Postel

Network Working Group
Request for Comments: 1025

J. Postel
ISI
September 1987

TCP AND IP BAKE OFF

Status of This Memo

This memo describes some of the procedures, scoring, and tests used in the TCP and IP bake offs held in the early development of these protocols. These procedures and tests may still be of use in testing newly implemented TCP and IP modules. Distribution of this memo is unlimited.

Introduction

In the early days of the development of TCP and IP, when there were very few implementations and the specifications were still evolving, the only way to determine if an implementation was "correct" was to test it against other implementations and argue that the results showed your own implementation to have done the right thing. These tests and discussions could, in those early days, as likely change the specification as change the implementation.

There were a few times when this testing was focused, bringing together all known implementations and running through a set of tests in hopes of demonstrating the N squared connectivity and correct implementation of the various tricky cases. These events were called "Bake Offs".

An early version of the list of tests included here appears in IEN-69 of October 1978. A demonstration of four TCP implementations was held at the Defense Communication Engineering Center in Reston, Virginia on 4 December 1978, and reported in IEN-70 of December 1978. A bake off of six implementations was held 27-28 January 1979 at USC-Information Sciences Institute in Marina del Rey, California and reported in IEN-77 of February 1979. And a distributed bake off was held in April 1980 over the network and reported in IEN-145 of May 1980.

The following section reproduces (with very slight editing) the procedure, tests, and scoring of the April 1980 Bake Off.

Procedure

This is the procedure for the TCP and IP Bake Off. Each implementor of a TCP and IP is to perform the following tests and to report the results. In general, this is done by using a test program or user Telnet program to open connections to your own or other TCP implementations.

Some test are made more interesting by the use of a "flakeway". A flakeway is a purposely flakey gateway. It should have control parameters that can be adjusted while it is running to specify a percentage of datagrams to be dropped, a percentage of datagrams to be corrupted and passed on, and a percentage of datagrams to be reordered so that they arrive in a different order than sent.

Many of the following apply for each distinct TCP contacted (for example, in the Middleweight Division there is a possibility of 20 points for each other TCP in the Bake Off).

Note Bene: Checksums must be enforced. No points will be awarded if the checksum test is disabled.

Featherweight Division

1 point for talking to yourself (opening a connection).

1 point for saying something to yourself (sending and receiving data).

1 point for gracefully ending the conversation (closing the connection without crashing).

2 points for repeating the above without reinitializing the TCP.

5 points for a complete conversation via the testing gateway.

Middleweight Division

2 points for talking to someone else (opening a connection).

2 points for saying something to someone else (sending and receiving data).

2 points for gracefully ending the conversation (closing the connection without crashing).

4 points for repeating the above without reinitializing the TCP.

10 points for a complete conversation via the testing gateway.

Heavyweight Division

10 points for being able to talk to more than one other TCP at the same time (multiple connections open and active simultaneously with different TCPs).

10 points for correctly handling urgent data.

10 points for correctly handling sequence number wraparound.

10 points for correctly being able to process a "Kamikaze" packet (AKA nastygram, christmas tree packet, lamp test segment, et al.). That is, correctly handle a segment with the maximum combination of features at once (e.g., a SYN URG PUSH FIN segment with options and data).

30 points for KOing your opponent with legal blows. (That is, operate a connection until one TCP or the other crashes, the surviving TCP has KOed the other. Legal blows are segments that meet the requirements of the specification.)

20 points for KOing your opponent with dirty blows. (Dirty blows are segments that violate the requirements of the specification.)

10 points for showing your opponents checksum test is faulty or disabled.

Host & Gateway IP Division

25 points for doing fragmentation and reassembly.

15 points for doing loose source route option.

15 points for doing strict source route option.

10 points for doing return route option.

10 points for using source quench messages.

10 points for using routing advice messages.

5 points for doing something with the type of service.

5 points for doing something with the security option.

5 points for doing something with the timestamp option.

5 points for showing that a gateway forwards datagrams without decreasing the time to live (showing a gateway is faulty).

5 points for showing that a gateway forwards datagrams with the time to live equal zero (showing a gateway is faulty).

10 points for showing that a gateway or hosts checksum test is faulty or disabled (showing a gateway is faulty).

Bonus Points

10 points for the best excuse.

20 points for the fewest excuses.

30 points for the longest conversation.

40 points for the most simultaneous connections.

50 points for the most simultaneous connections with distinct TCPs.

Tests

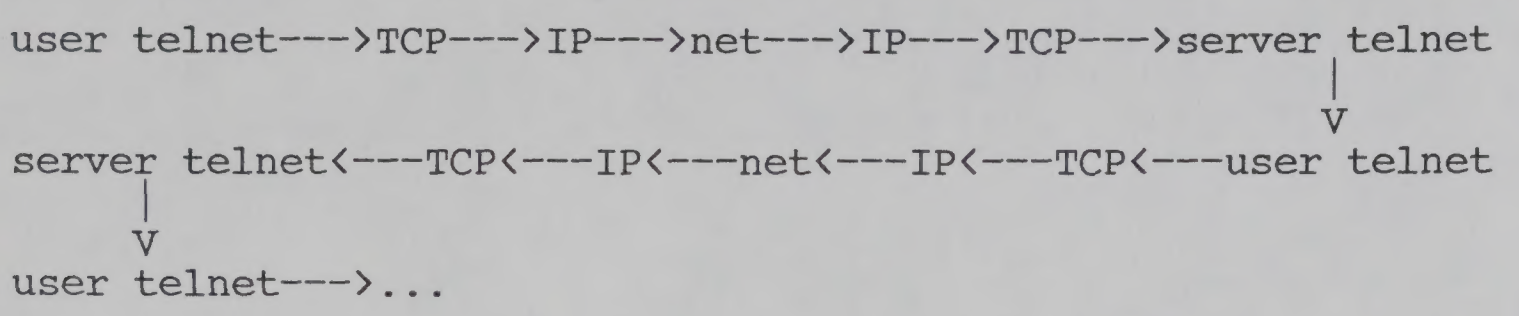
The following tests have been identified for checking the capabilities of a TCP implementation. These may be useful in attempting to KO an opponent.

1. Single connection. Open & close a single connection many times.

2. Multi connections. Open several connections simultaneously. Two connections to the same socket (i.e., a-b and a-c) check proper separation of data.

3. Half Open Connection. Open a connection, crash local TCP and attempt to open same connection again.

4. Piggy-back Loop. Open connections via Telnet.



5. Maximum connections. Open connections between a pair of TCP until refused or worse.

6. Refused connection. Open a connection to a non-accepting socket, does it get refused?

7. Zero Window. Try to send data to a TCP that is presenting a zero window.

8. Fire Hose. Make many connections to data source ports, or connections to a data sink and send as fast as you can.

9. Urgent Test. Try to send data to a user program that only receives data when in urgent mode.

10. Kamikazi Segment. Send and receive nastygrams. A nastygram is a segment with SYN, EOL, URG, and FIN on and carrying one octet of data.

11. Sequence Wraparound. Test proper functioning when sequence numbers (a) pass 2**31 (i.e., go from plus to "minus") and (b) pass 2**32 (i.e., go from 2**32-1 to 0).

12. Buffer size. With buffer size not equal to one, send data in segments of various sizes, use urgent occasionally.

13. Send a nastygram into a half open connection when the sequence number is about to wrap around.

New Ideas

The above tests check for basic operation and handling of some of the tricky cases. They do not consider performance in any way, or check to see if some of the recently developed ideas have been implemented.

New Mechanisms

1. The John Nagel Procedures (RFC-896).

2. The Van Jacobson Procedures (slow start, RTT measurements, etc).

3. The SQuID Procedures (RFC-1016).

Performance Tests

Performance tests are difficult to specify because the results depend so much on the state of the environment of the test. Here are a few possibilities:

1. FTP Throughput: Send a 1 megabyte file to a locally nearby machine on an Ethernet measuring the elapsed time.

2. FTP Throughput: Send a 1 megabyte file to a locally nearby machine on an ARPANET measuring the elapsed time.

3. NETBLT Throughput: Send a 1 megabyte file to a locally nearby machine on an Ethernet measuring the elapsed time.

4. NETBLT Throughput: Send a 1 megabyte file to a locally nearby machine on an ARPANET measuring the elapsed time.

5. Character Test: Use a test program to send a character via TCP to the Echo Server (RFC-862), time the round trip (from the time the character is sent until the echo is returned to the test program).

Appendix

For History Buffs Only:

The following item was in the original 1980 tests, but has been moved to this appendix since it no longer applies.

10 points for correctly handling rubber baby buffer bumpers in both directions (End of Letter sequence number adjustments).